Natural Language Processing Assignment - Sentiment Analysis

Abil N George(CS13M002), Bharadwaj J(CS13M058), and Anil Kumar(CS13M004)

Dept. of CSE, IIT Madras

Abstract. Sentiment Analysis, also referred to as Opinion Mining attempts to capture, what an author of a document opines or feels, for example, what a film reviewer thinks about a film, what an online shopping consumer thinks about the product that he has purchased, etc. If we have a knowledge of, what kind of words or phrases convey a positive sentiment or a good sentiment, then the problem becomes relatively easy. Just a mere search, would be sufficient. But, the real challenge is when the system itself has to 'learn', which words convey positive sentiments and which words convey negative sentiments.

1 Introduction

Sentiment analysis is essentially, a supervised learning technique for classifying and/or rating text documents or resources. That is, a labelled training data is available. Using this data, the program must learn how to classify a new document, what are the essential features that discriminate one class of documents from another, how to extract them, etc.

Hence, we make use of a labelled corpus (i.e **Polarity Dataset v3.0 available at http://www.cs.cornell.edu/- people/pabo/movie-review-data/**) in training the system. A set of 1000 positive movie reviews and 1000 negative movie reviews were used to train the system. NLTK toolkit was used for the various text processing involved. Once trained, the classifier must be able to differentiate between good and bad movie reviews.

2 Literature Survey

Our work, simply attempts to classify the movies as good or bad (i.e. a two class classification problem). A more challenging problem is to classify certain reviews as 'neutral'. In general, most of the movies fall under this category and hence it would be ideal to have a neutral class. But a simple extension to 3 - class classification alone might not be sufficient. One of our team members, had done this problem as part of the Machine Learning course. The biggest challenge was to get a good accuracy for the neutral class. While, the other classes continued to give good results, the neutral class results were not satisfactory. It involves assigning misclassification costs and boosting mechanisms for achieving a good accuracy.

An even complicated approach is to rate the movies in a scale from 1-5. This was proposed in [2]. Apart from Artificial Intelligence, this method also requires collective Human Intelligence, as it requires us to meet different persons and collect opinions on movie reviews and observe how well they are able to differentiate between two differently rated movie reviews in the corpus. Also, this will involve regression rather than classification for getting accurate ratings.

3 Methodology

The entire process of Sentiment Analysis can be divided into the following four stages: *Document-Vector matrix creation, Sentiment words extraction (or feature extraction), Training classifiers on the reduced data, Testing on new data.*

3.1 Document-Vector model

Document-Vector model is used to represent the presence or frequency of words in a set of documents. It is called bag-of-words representation, where the rows of a matrix define a document and the columns represent the words in the document. The cell values can be either 0/1 indicating absence or presence of words in documents or they can also hold the frequency of words in documents. Though, the frequency based bag-of-words model works well for certain NLP applications, for Sentiment analysis the 0/1 matrix gives better results. The reason for the same is intuitive: a frequent word need not necessarily be a sentiment word. For example, in a movie review irrelevant words like *'actor', 'scene', 'music' etc* can have more frequency than actual sentiment words like *'breathtaking', 'stunning', 'astounding' etc.* Hence, we have employed the 0/1 based approach.

3.2 Feature Selection

Now, that we have a good representation of data, we need to make it better by eliminating features that will not be making any contribution to the overall sentiment od the data. The examples mentioned above i.e. words like *'breathtaking', 'stunning', 'astounding'* need to be retained and rest be removed. There are various feature reduction techniques. We have attempted the following methods: *Categorical Proportional Difference and CHI-squared* (χ^2) *feature extraction.*

Categorical Proportional Difference: This method is based on the assumption that, if a particular word occurs in approximately equal number of times in any of the 2 classes, then *that feature does not have the power to classify or differentiate between those 2 classes.* Hence such features need to be removed. According to *Bo Pang et. al* [1], for a 2 class problem it can be expressed as follows:

$$PD = \frac{|n(f_i^+) - n(f_i^-)|}{n(f_i^+) + n(f_i^-)} \qquad (1)$$

where
$n(f_i^+)$: number of occurrences of i^{th} feature in positively labelled documents
$n(f_i^-)$: number of occurrences of i^{th} feature in negatively labelled documents.
We are selecting the top 2000 features and rejecting the others.

CHI Squared (χ^2) method: This method determines whether two events, E and F are independent of each other [3]. In this case, the events are the feature to be selected and the output class labels. It is calculated as follows:

$$\chi^2(t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} \frac{(N_{e_t,e_c} - E_{e_t,e_c})^2}{E_{e_t,e_c}} \tag{2}$$

where
N_{e_t,e_c}: Total number of co-occurrences of term t in class c
E_{e_t,e_c}: Expected number. of co-occurrences of term t in class c.

Since, the value of χ^2 implies independence, features that lead to high values are independent of the class output and these can be rejected. Just like PD method, we are selecting the top 2000 features and rejecting the others

3.3 Training and Testing

For training and testing data, we can divide the actual data set into two unequal sets in such a way that the larger data set can be used for training and the remaining for testing. Also it is required to cross-validate the left out data. Hence each time we split the data we shuffle the data and then select the train and test data sets.

On the train data, we need to learn classifiers. The following classifiers were trained: *Naive Bayes and Support vector Machines*

Naive Bayes method: It is based on the assumption that each feature is independent of the other features. The classification can be done by just applying the Bayes rule after considering the Naive Bayes assumption [3] as follows:

$$C_i = argmax_c P(c|d_i) = argmax_c (P(c|f_{i1}).P(c|f_{i2}).P(c|f_{i3})...P(c|f_{in})) \tag{3}$$

This directly gives the most probable class for the i^{th} document. Though, the method appears very simple compared to the other classification techniques, the Naive Bayes classifier is known to give surprisingly very good results for text based classifications. Out experiment is also no different.

Support Vector Machines: Support Vector Machines are based on perceptron based classifying methods, that consider the vectors as points in space and attempt to maximize a distance between the classes while allowing some outlier points alone to crossover i.e. slack. A detailed discussion on how the SVM classifier works is beyond the scope of this experiment. The only reason we attempted this was to compare the Naive Bayes results with the results of a proven efficient classifier. The results showed us that the Naive Bayes classifier performed marginally better.

4 Experiments and Results

Table 1. Naive Bayes Classification results with PD feature selection

	Accuracy	Positive			Negative		
		Precision	Recall	F-measure	Precision	Recall	F-measure
0	0.88500000	0.92307692	0.84000000	0.87958115	0.85321101	0.85321101	0.88995215
1	0.89500000	0.92473118	0.86000000	0.89119171	0.86915888	0.86915888	0.89855072
2	0.88500000	0.95294118	0.81000000	0.87567568	0.83478261	0.83478261	0.89302326
3	0.88000000	0.95238095	0.80000000	0.86956522	0.82758621	0.82758621	0.88888889
4	0.89000000	0.93333333	0.84000000	0.88421053	0.85454545	0.85454545	0.89523810
5	0.89500000	0.93406593	0.85000000	0.89005236	0.86238532	0.86238532	0.89952153
6	0.87500000	0.93103448	0.81000000	0.86631016	0.83185841	0.83185841	0.88262911
7	0.88000000	0.88775510	0.87000000	0.87878788	0.87254902	0.87254902	0.88118812
8	0.88000000	0.92222222	0.83000000	0.87368421	0.84545455	0.84545455	0.88571429
9	0.91000000	0.95555556	0.86000000	0.90526316	0.87272727	0.87272727	0.91428571
avg	0.88750000	0.93170969	0.83700000	0.88143220	0.85242587	0.85242587	0.89289919

Table 2. SVM classifcation results with PD feature selection

	Accuracy	Positive			Negative		
		Precision	Recall	F-measure	Precision	Recall	F-measure
0	0.85500000	0.88172043	0.82000000	0.84974093	0.83177570	0.83177570	0.85990338
1	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000
2	0.90000000	0.90816327	0.89000000	0.89898990	0.89215686	0.89215686	0.90099010
3	0.83000000	0.85869565	0.79000000	0.82291667	0.80555556	0.80555556	0.83653846
4	0.86000000	0.85294118	0.87000000	0.86138614	0.86734694	0.86734694	0.85858586
5	0.89500000	0.89898990	0.89000000	0.89447236	0.89108911	0.89108911	0.89552239
6	0.86000000	0.86000000	0.86000000	0.86000000	0.86000000	0.86000000	0.86000000
7	0.84000000	0.83333333	0.85000000	0.84158416	0.84693878	0.84693878	0.83838384
8	0.85500000	0.83177570	0.89000000	0.85990338	0.88172043	0.88172043	0.84974093
9	0.89500000	0.86915888	0.93000000	0.89855072	0.92473118	0.92473118	0.89119171
avg	0.86400000	0.86447783	0.86400000	0.86375443	0.86513146	0.86513146	0.86408567

Table 3. Naive Bayes Classification results with χ^2 feature selection

	Accuracy	Positive			Negative		
		Precision	Recall	F-measure	Precision	Recall	F-measure
0	0.92500000	0.94736842	0.90000000	0.92307692	0.90476190	0.90476190	0.92682927
1	0.88500000	0.88118812	0.89000000	0.88557214	0.88888889	0.88888889	0.88442211
2	0.90500000	0.91752577	0.89000000	0.90355330	0.89320388	0.89320388	0.90640394
3	0.87500000	0.87878788	0.87000000	0.87437186	0.87128713	0.87128713	0.87562189
4	0.91000000	0.92708333	0.89000000	0.90816327	0.89423077	0.89423077	0.91176471
5	0.91000000	0.91836735	0.90000000	0.90909091	0.90196078	0.90196078	0.91089109
6	0.89500000	0.90721649	0.88000000	0.89340102	0.88349515	0.88349515	0.89655172
7	0.87500000	0.85714286	0.90000000	0.87804878	0.89473684	0.89473684	0.87179487
8	0.87000000	0.92045455	0.81000000	0.86170213	0.83035714	0.83035714	0.87735849
9	0.89500000	0.93406593	0.85000000	0.89005236	0.86238532	0.86238532	0.89952153
avg	0.89450000	0.90892007	0.87800000	0.89270327	0.88253078	0.88253078	0.89611596

Table 4. Naive Bayes Classification results with χ^2 feature selection

	Accuracy	Positive			Negative		
		Precision	Recall	F-measure	Precision	Recall	F-measure
0	0.89000000	0.86111111	0.93000000	0.89423077	0.92391304	0.92391304	0.88541667
1	0.87000000	0.87000000	0.87000000	0.87000000	0.87000000	0.87000000	0.87000000
2	0.85500000	0.86597938	0.84000000	0.85279188	0.84466019	0.84466019	0.85714286
3	0.83500000	0.83838384	0.83000000	0.83417085	0.83168317	0.83168317	0.83582090
4	0.85500000	0.83177570	0.89000000	0.85990338	0.88172043	0.88172043	0.84974093
5	0.86500000	0.86868687	0.86000000	0.86432161	0.86138614	0.86138614	0.86567164
6	0.80500000	0.82105263	0.78000000	0.80000000	0.79047619	0.79047619	0.80975610
7	0.87500000	0.86407767	0.89000000	0.87684729	0.88659794	0.88659794	0.87309645
8	0.85000000	0.88043478	0.81000000	0.84375000	0.82407407	0.82407407	0.85576923
9	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000	0.85000000
avg	0.85500000	0.85515020	0.85500000	0.85460158	0.85645112	0.85645112	0.85524148

5 Conclusions and Future Work

From the above results, it is clear that, Naive Bayes classification using χ^2 feature selection gives the best results. The bag of words model used, considers features independently. Even if the words were rearranged in a document we will still get similar results as long as the words are present. This is a serious shortcoming. For example, consider the sentence 'This is not a good movie'. The word 'good' independently adds positivity to the document. To eliminate this we applied a method followed in [1] where the words following negative words like 'not' were appended with '_NOT' till end of sentence is reached. However the results did not improve by much.

A better alternative for the 0/1 model, is to add SentiWordNet scores. This is also proposed by Bo Pang [1] with a modified formula for calculating PD. This gives more information about the sentiment of a term and hence might work effectively.

References

[1] Bo Pang, Lillian Lee Shivakumar Vaithyanathan: Thumbs up? Sentiment Classification using Machine Learning Techniques (2002).

[2] Bo Pang and Lillian Lee: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales

[3] Christopher D. Manning and Hinrich Schütze: Foundations of Statistical Natural Language Processing (1999)